PUC-Rio
Departamento de Informática
Período: 2015.1
Horário: 2as-feiras e 4as-feiras de 17-19
20 de maio de 2015

ESTRUTURAS DE DADOS AVANÇADAS (INF 1010)

2^a Lista de Exercícios

1. **Lista**

   (a) Seja um TAD definido por uma lista circular implementada em um vetor.

      i. Dado que o vetor possui max posições, proponha uma função que determine o número n de elementos a partir do conteúdo dos ponteiros de início H e final T da lista.

      ii. Proponha um algoritmo para determinar o k-ésimo elemento da lista. Analise sua complexidade.

      iii. Proponha um algoritmo para remover o k-ésimo elemento da lista. Qual o número total de operações? E a sua complexidade?

      iv. Proponha um algoritmo para inserir um elemento x logo após o k-ésimo elemento da lista. Analise sua complexidade e o número total de operações.

   Árvores

   (a) Qual a maior e menor quantidade de nós que podem existir em uma árvore binária completa de altura h ?

   (b) Uma árvore binária cuja altura é igual ao número de nós pode possuir nós cheios ? Justifique.

   (c) Toda árvore binária cheia é completa ? Justifique.

   (d) Represente a sequência abaixo na forma de árvores binárias de alturas mínima e máxima.
   s = { 3, 5, 9, 12, 14, 6, 7, 15 }

   (e) Desenhe uma árvore estritamente binária com 7 nós para a qual os percursos em pré-ordem e in-ordem produzem a mesma sequência de visitas.

   (f) Dados os percursos abaixo, reconstruir a árvore original:
   pré-ordem: 1, 2, 3, 6, 8, 4, 9, 10, 12, 11, 5, 7, I
   simétrica (in-ordem) : 6, 3, 8, 2, 4, 9, 12, 10, 11, 1, I, 7, 5

2. Responda Certo ou Errado, justificando.

(a) Qualquer que seja o número de chaves, é sempre possível construir com elas uma árvore binária completa.

(b) Qualquer que seja o número de chaves, é sempre possível construir com elas uma árvore binária cheia.

(c) Uma árvore binária que possui as folhas no último ou penúltimo níveis é completa.

(d) Dada uma árvore binária com mais de 3 nós, é possível que um percurso em pré-ordem e um percurso em ordem simétrica visitem os nós na mesma ordem ?

(a) Utilizando uma árvore binária para representar uma expressão aritmética.

i. Represente, em uma árvore binária, a seguinte expressão aritmética, respeitando a precedência usual dos operadores: 2 * (5 – 3) + (2 + 6) / 4. As folhas devem corresponder aos operandos e os nós interiores aos operadores. Não represente os parênteses.

ii. Defina, em C, a estrutura do nó da árvore binária utilizada para representar expressões. Assuma que os operandos são inteiros positivos e que apenas os operadores binários +, -, * e / são aceitos. A estrutura deve possuir o menor número possível de campos.

iii. Escreva em C uma função recursiva que recebe como parâmetro o endereço do nó raiz de uma árvore binária representando uma expressão e retorna como resultado o valor obtido na avaliação da expressão. Utilize a estrutura do nó definida no item c.

iv. Qual o percurso implementado pela função escrita no item anterior? O que realiza o procedimento de visita ?

i. Verificar se as árvores abaixo são binárias de busca

1

ii. Construir algoritmo para dada uma árvore n-ária, transformá-la em uma árvore binária.

iii. Escrever rotina em pseudo-código e em C para percorrer uma árvore binária qualquer, em nível, das folhas para a raiz. Indique a complexidade do algoritmo.

iv. Escrever uma rotina em C para buscar a informação em um nó de uma árvore binária de busca, sendo dada a sua chave. Cuide para que a rotina seja eficiente.

v. Achar o maior elemento (campo numérico) de uma árvore binária dada.

i. Uma árvore binária é *zig-zag* quando é vazia ou quando **não** possui nós cheios. Sem usar recursividade, escreva, em C, uma função que recebe o endereço do nó raiz de uma árvore binária e verifica se ela é zig-zag. Se a árvore dada for zig-zag, a função deverá retornar sua altura; caso contrário, deverá retornar –1. **Não** utilize estruturas de dados auxiliares (pilhas, filas, etc...); apenas variáveis locais dos tipos ponteiro ou inteiro.

i. Escreva, em C, uma função recursiva que permuta as subárvores esquerda e direita de todos os nós de uma árvore binária. A função deve receber como parâmetro o endereço do nó raiz da subárvore a ser processada. Explicite também a chamada externa.

i. Considere uma árvore binária de busca implementada com a seguinte estrutura de nó:

```
typedef struct bst_node BstNode;
struct _bst_node {
void* info;
AvlNode* parent;
AvlNode* left;
AvlNode* right;
};
```

Escreva a função que determina o antecessor de um dado nó, visitando o menor número de nós possível e que tenha o seguinte protótipo:

```
BstNode* prev_node(BstNode* node);
```

Resp:

```
BstNode* max (BstNode* r) {
if (r==NULL) return NULL;
while(r->right!=NULL) r=r->right;
return r;
}
BstNode* prev_node(BstNode* node) {
if (node==NULL) return NULL;
else if (node->left!=NULL) { /* return max(node->left); */
node = node->left;
while (node->right!=NULL) node=node->right;
return node;
}
else { /* retorna o ancestral */
BstNode* p=r->parent;
while (p->parent!=NULL && node==p->left ) {
node = p;
p = p->parent;
}
return p;
}
}
```